CLAIMS

1. A method for frequency-offset error determination, comprising the steps of:

receiving a string composed of a same basic constituent repeated ten times, where a basic constituent is generated from a sequence, defined in the frequency domain, containing QPSK-like modulated elements;

sampling said string to collect a plurality of measurements each including two of said basic constituents;

accounting for all measured differences amongst said plurality of samples as being solely attributable to a common frequency-offset error; and

correcting for said common frequency-offset error in later digital signal processing.

2. The method of Claim 1, wherein:

the step of sampling includes skipping a first few of said repeating QPSK symbols before a first measurement is taken.

3. The method of Claim 1, wherein:

the step of sampling includes measuring said string of repeating QPSK-like modulated elements three times to obtain $s_1(n)$, $s_2(n)$, and $s_3(n)$, ; and

the step of accounting sets up for solution a matrix,

$R = Z^H Z$ with,

$$Z = \begin{bmatrix} s_1(n) & s_2(n) & s_3(n) \end{bmatrix}$$

$$s_1(n) = \sigma_1(n) + \eta_1(n)$$

$$s_2(n) = \sigma_2(n) + \eta_2(n)$$

$$s_3(n) = \sigma_3(n) + \eta_3(n)$$

$$\sigma_1(n) = Ae^{j2\pi\frac{v}{F_s}n + j\Phi(n) + j\varphi}$$

$$\sigma_2(n) = Ae^{j2\pi\frac{v}{F_s}(n+N) + j\Phi(n+N) + j\varphi} = e^{j2\pi\frac{v}{F_s}N}\sigma_1(n)$$

$$\sigma_3(n) = Ae^{j2\pi\frac{v}{F_s}(n+2N) + j\Phi(n+2N) + j\varphi} = e^{j2\pi\frac{v}{F_s}N}\sigma_2(n) = e^{j4\pi\frac{v}{F_s}N}\sigma_1(n)$$

and that can be expanded to,

$$R=\begin{bmatrix} MA^2+\gamma_{1,1}^M & MA^2e^{j2\pi\frac{v}{F_s}N}+\gamma_{1,2}^M & MA^2e^{j4\pi\frac{v}{F_s}N}+\gamma_{1,3}^M \\ MA^2e^{-j2\pi\frac{v}{F_s}N}+\overline{\gamma}_{1,2}^M & MA^2+\gamma_{2,2}^M & MA^2e^{j2\pi\frac{v}{F_s}N}+\gamma_{2,3}^M \\ MA^2e^{-j4\pi\frac{v}{F_s}N}+\overline{\gamma}_{1,3}^M & MA^2e^{-j2\pi\frac{v}{F_s}N}+\overline{\gamma}_{2,3}^M & MA^2+\gamma_{3,3}^M \end{bmatrix}$$

with:

$$\gamma_{k,l}^M=\sum_{n=1}^{M}\eta_k(n)\tilde{\eta_l}(n)$$

wherein, in the absence of noise, R is rank 1 and decomposes as,

$$R=\sum_{n=1}^{K}\lambda_n a_n^H a_n$$

$$-x=\sum_{m=1}^{K}\gamma_m a_m$$

4. The method of Claim 3, wherein:

the step of accounting recognizes any received signals is adversely affected by additive white Gaussian noise and multi-path interference, and determines R's maximum eigenvector in a first step followed by an iterative method with two iterations for a pseudo rank one matrix, and generally conforms to, expressing R and any vector $x\in C^K$ using the eigenvector basis,

$$R=\sum_{n=1}^{K}\lambda_n a_n^H a_n$$

$$x=\sum_{m=1}^{K}\gamma_m a_m$$

wherein, the vector matrix products are computed by,

$$x_1 = x_0 R = \sum_{m=1}^{K} \gamma_m a_m \sum_{n=1}^{K} \lambda_n a_n^H a_n = \sum_{m=1}^{K}\sum_{n=1}^{K} \gamma_m \lambda_n \left(a_m a_n^H\right) a_n = \sum_{m,n=1}^{K} \gamma_m \lambda_n \delta_{m\ n} a_n = \sum_{m=1}^{K} \gamma_m \lambda_m a_m$$

$$x_2 = x_1 R = \sum_{m=1}^{K} \gamma_m \lambda_m a_m \sum_{n=1}^{K} \lambda_n a_n^H a_n = \sum_{m=1}^{K}\sum_{n=1}^{K} \gamma_m \lambda_m \lambda_n \left(a_m a_n^H\right) a_n = \sum_{m,n=1}^{K} \gamma_m \lambda_m \lambda_n \delta_{m,n} a_n = \sum_{m=1}^{K} \gamma_m \lambda_m^2 a_m$$

and this iterative operation is repeated up to x_k,

$$x_k = x_{k-1} R = \sum_{m=1}^{K} \gamma_m \lambda_m^{k-1} a_m \sum_{n=1}^{K} \lambda_n a_n^H a_n = \sum_{m=1}^{K}\sum_{n=1}^{K} \gamma_m \lambda_m^{k-1} \lambda_n \left(a_m a_n^H\right) a_n = \sum_{m,n=1}^{K} \gamma_m \lambda_m^{k-1} \lambda_n \delta_{m,n} a_n = \sum_{m=1}^{K} \gamma_m \lambda_m^k a_m$$

.

5. The method of Claim 3, wherein:

the step of accounting assumes R to have pseudo rank one, and the spectrum of eigenvalues is such that $\lambda_1 = \lambda_1 >> \lambda_2, \lambda_3, ..., \lambda_K$, so x_k rapidly converge to $\lambda_1^k a_1$ as k increases.

6. The method of Claim 1, further comprising the steps of:

computing f(μ) for sixty-four equally spaced values of μ ranging from 0 to 208.33kHz once a maximum eigenvector is available; and

identifying a bulky peak which provides $\hat{v}$, the frequency offset estimate.

8. The method of Claim 1, further comprising the steps of:

determining the sign of the imaginary part of the second component of the dominant eigenvector;

based upon the sign of the imaginary part of the second component of the dominant eigenvector, conjugate (or not) 64 pre-stored steering vectors which span possible frequency offsets from 0 to 208.33kHz;

computing an objective function, f(υ) over requency offset in the range of 0 to 208.33kHz; when μ = v (true frequency offset), then a maximum of occurs;

finding anthe index of the maximum of f(μ);

using this index as an element into a pre-stored lookup table consisting of $e^{j2\pi(\frac{\mu_i}{F_s})}$ for all μ_i from 0 to 208.33kHz in 64 increments; where single values of the complex exponential are stored for all positive frequency offsets;

let an index of the maximum of f(μ) = j;

then, a value retrieved from the table is $e^{j2\pi(\frac{\mu_J}{F_s})}$;

based upon a sign of the imaginary part of a second component of a dominant eigenvector, conjugate this value to get a proper starting point for a frequency-correcting cisoid;

wherein remaining elements of said frequency-correcting cisoid are generated as follows:

$e^{j2\pi(\frac{\mu_J}{F_s})}$ is retrieved from a table and stored to memory at memory location X;

loading a resulting value from memory, multiplied with itself, producing $e^{j2\pi(\frac{\mu_J}{F_s})2}$;

storing a resulting value to memory location X+1;

(vector) loading the two values of the correction cisoid from memory location X into the processor;

wherein a last value of this vector is multiplied by the entire vector to produce next samples in a correction cisoid;

storing these two values to memory location X+2.

$$\begin{bmatrix} e^{j2\pi(\frac{\mu_J}{F_s})3} & e^{j2\pi(\frac{\mu_J}{F_s})4} \end{bmatrix} = e^{j2\pi(\frac{\mu_J}{F_s})2} * \begin{bmatrix} e^{j2\pi(\frac{\mu_J}{F_s})} & e^{j2\pi(\frac{\mu_J}{F_s})2} \end{bmatrix};$$

Reloading this vector into the processor starting at memory location X with length=4; and repeating this procedure 4 more times to obtain a correction cisoid.